AX1003 DAC 的使用



AN-AX1000-007-V10-CH-DAC

2006 年 8 月

版本 1.0

1 简介

AX1003内置了一个10Bit R2R DAC，I/O功能上与RA2复用。此DAC是一个电阻网络分压形式的架构，其参考电压为AVDD。输出端的计算公式为：

DAC_OUT = AVDD x (DIN / 1024) （DIN为10bit输入数据）

DAC的输入寄存器和ADC的输出寄存器共用两个地址，分别是DACH（41H）和DACL（42H），当对这两个寄存器进行读操作时，读出的是ADC的采样结果；当向这两个寄存器写入数据时，相当于是将数据写入DAC的BUF中。AX1003的DAC只有10Bit精度，DACH中8Bit为高8位，DACL中只有高两Bit有效，为低两位，Bit0~Bit5会被自动丢掉。

2 触发模式

用户的程序将数据写入DACH和DACL中是有一个先后顺序的，当程序将数据写入DACH和DACL时，其结果并不是立即会输出到RA2上，何时输出，要根据所选择的DAC输出触发模式来决定。DAC的输出有两种触发模式，一种是写入触发模式，另外一种是时间触发模式。这两种模式的选择由DACTL（ADCCFG2，5）来决定。

3 写入触发模式

当DACTL=0时，选择写入触发模式，触发方式是当向DACH写入数据的动作完成后，马上从RA2上输出DAC的结果，所以在此种模式下工作时，为了保证输出的是想要得到的结果，一定要先将数据写入DACL后，再写入DACH的数据，芯片复位时，默认选择此工作模式。

4 定时器触发模式

当DACTL=1时，选择定时器触发模式，可以通过DACTC（ADCCFG1，5）来选择触发定时器为Timer1或Timer2，在定时器触发模式下，每当所选的定时器有中断发生时，cpu会自动将DACH和DACL中的数据输出到RA2上。为了保证输出正确，用户程序一定要保证在定时器的中断到来之前更新DACH和DACL中的数据。

第一种触发模式比较适合于用在调节控制系统中，当要改变输出结果时，只需要向DACH和DACL中写入数据，输出结果马上就可以被改变；而第二种触发模式比较适合于用在定时输出系统中，例如音频输出。

5 输出缓冲

为了提高DAC的输出能力，方便与外接的放大电路进行匹配，在DAC的输出中有一级缓冲，用户可以通过设置DACBE（ADCCFG2，6）来使能/禁止DAC缓冲。

6 DAC 配置步骤

AX1003的DAC模块配置方法如下：

1、 设置RA2为输入
2、 选择DAC的触发模式
3、 如果选择了定时器触发模式，则需要对相关的定时器时行初始化
4、 选择是否使用DAC输出缓冲
5、 使能DAC

完成配置后，就可以向DAC中写入数据了。

与DAC相关的寄存器如下表：

寄存器名称	寄存器地址	与 DAC 的相关位
ADCCFG1	03FH	DACTC(Bit5)
ADCCFG2	043H	DACEN(Bit7),DACBE(Bit6),DACTL(Bit5)
ADCH	041H	
ADCL	042H	

ADCCFG1 register (address: 03Fh, R/W)

ADCBE	ADCTC	DACTC	ADCBSEL	RESERVED	ADCISEL2	ADCISEL1	ADCISEL0
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	0	0	0	-	1	1	1

Bit 7: **ADCBE**: ADC input buffer enable
0 = disable ADC input buffer
1 = enable ADC input buffer

Bit 6: **ADCTC**: ADC timer trigger mode timer select
0 = Select timer1
1 = Select timer2

Bit 5: **DACTC**: DAC timer trigger mode timer select
0 = Select timer1
1 = Select timer2

Bit 4: **ADCBSEL**: ADC resolution selection
0 = 10-bit resolution
1 = 8-bit resolution

Bit 3: **RESERVED**: Reserved bit

Bit 2-0: **ADCSEL2**: **ADCSEL0**: Analog input select bit
000 = RB0
001 = RB1
010 = RB2
011 = RB3
100 = RB4
101 = RB5
110 = RB6
111 = Bandgap Vref

ADCCFG2 register (address: 043h, R/W)

DACEN	DACBE	DACTL	ADCTL	ADCRFSEL	ADCGO	ADCDONE	ADCIE
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	0	0	0	0	0	0	0

Bit 7: **DACEN**: DAC enable
0 = disable DAC
1 = enable DAC

Bit 6: **DACBE**: DAC output buffer enable
0 = disable DAC output buffer
1 = enable DAC output buffer

Bit 5: **DACTL**: DAC Timer trigger mode selection
0= disable DAC timer trigger mode
1 = enable DAC timer trigger mode

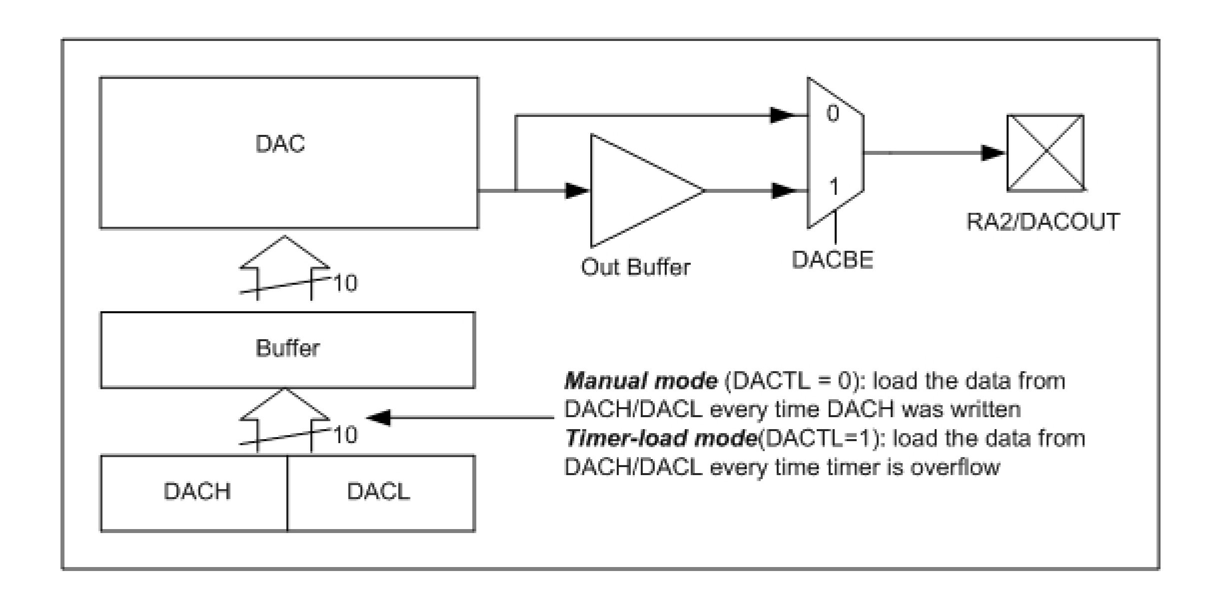
Bit 4: **ADCTL**: ADC Timer trigger mode selection
0 = disable ADC timer trigger mode
1 = enable ADC timer trigger mode

Bit 3: **ADCRFSEL**: Reference voltage input select bit
0 = AVDD is selected
1 = RB7 is selected

Bit 2: **ADCGO**: ADC GO/DONE bit
0 = When the conversion has completed, this bit is read as 0
1 = Set 1 to start the conversion. While the conversion is in progress, this bit is read as 1

Bit 1: **ADCDONE**: ADC interrupt pending bit
0 = no interrupt happens
1 = interrupt happens

Bit 0:	**ADCIE**: ADC interrupt enable 0 = ADC interrupt is not enabled 1 = ADC interrupt is enabled



7 源代码例子

以下程序将以 Timer2 触发的方式使 DAC 产生 1KHz 的正弦波

```
;Crystal input = 16M

#include "ax1003.inc"

    DATASEG
    OutputFlag        equ        100h
    OutputCounter     equ        101h
    SOUND_WAVE_H      equ        102h
    SOUND_WAVE_L      equ        103h

    CODESEG

    org      0fffh                ;ISD mode
    jmp      MAIN

    org      002h            ;entrence of timer2 interrupt
    jmp      TIMER2_ISR

    org      010h
;*************************************
;*           MAIN PROGRAM           *
;*************************************
MAIN:
    call     CONFIG_PLL           ;Config PLL=48MHz
    call     CONFIG_DAC           ;config DAC,timer2 trigger mode
    call     CONFIG_TIMER2        ;config timer2
MAIN_LOOP:
    Jmp      MAIN_LOOP


CONFIG_PLL:
    movw 18
    mov      PLL_NP, w
    movw     0x02
    mov      PLL_NR, w     ;set the clock at 96MHz
    movw     0x12
    mov      CCR, w

    jbs      CCR,7
    jmp      $-1
```

```
     bs        CCR,5         ;set PLL as system clock
     nop
     nop
     nop
     nop
     nop
     nop
     nop
     ret

CONFIG_DAC:
     bs       RADIR,2        ;set RA2 input
     bs       DACTC          ;select timer2 to trigger DAC output
     bs       DACTL          ;enable DAC timer trigger mode
     bs       DACEN          ;enable DAC module
     clr      OutputFlag
    clr      OutputCounter
     ret
CONFIG_TIMER2:
    movw  124            ;set timer2 Compare value
     mov       TMR2_PR,w
     movw      00000100B      ;16 divi
     mov       TMR2_PSR,w     ;timer2 interrupt cycle = 48000K/16/(124+1) = 24K
     bs        TMR2IE
    bc      GIE
    bs      tmr2on
     ret

TIMER2_ISR:
     bc         TMR2OF        ;clear timer2 interrupt pending bit


     clr        mach
     movw       0xf0
     mov        macw,w
     mov        w,OutputCounter
     add        macw,w
     movw       0x03
     addc       mach,w
     IREAD
     MOV       SOUND_WAVE_L, W   ;delet low six bits
      mov      w,h
      mov      SOUND_WAVE_h,w
      JBS        SOUND_WAVE_h,7
      JMP      positive_
      not      SOUND_WAVE_L
      inc      SOUND_WAVE_L
      jbc      z
      dec      SOUND_WAVE_h
      not      SOUND_WAVE_h
      movw         0xc0
      and      SOUND_WAVE_L,w
      not      SOUND_WAVE_L
      inc      SOUND_WAVE_L
      jbc      z
      dec      SOUND_WAVE_h
      not      SOUND_WAVE_h

      jmp      outaudio_data_exit1
positive_:
      movw     0xc0
      and      SOUND_WAVE_L,w
outaudio_data_exit1:
      movw     0x80                ;convert singed to unsigned
      add      SOUND_WAVE_h,w
     nop
      MOV      W,SOUND_WAVE_L
      MOV      DACL,W

     MOV       W,SOUND_WAVE_H
      MOV      DACH,W

     inc        OutputCounter
     movw       24
     xor        w,OutputCounter
     jbs        z
     reti
```

```
    clr       OutputCounter          ;start a new cycle

    reti

;sine wave table,
org  0x3f0
DB 0x00,0x00,0x3E,0x06,0x0E,0x0C,0x0D,0x11,0xE2,0x14,0x4A,0x17,0x1C,0x18,0x4B,0x17
DB 0xE1,0x14,0x0C,0x11,0x0E,0x0C,0x3E,0x06,0x00,0x00,0xC2,0xF9,0xF2,0xF3,0xF3,0xEE
DB 0x1F,0xEB,0xB6,0xE8,0xE4,0xE7,0xB7,0xE8,0x1F,0xEB,0xF3,0xEE,0xF2,0xF3,0xC2,0xF9
```

销售与支持

卓荣集成电路科技有限公司
香港新界沙田香港科学园科技大道西 6 号集成电路开发中心 1 楼 110-111 室
电话: +(852) 2607 4090
传真: +(852) 2607 4096
电邮: info@appotech.com

建荣集成电路科技(珠海)有限公司
广东省珠海市吉大白莲路 184 号立体科技大厦 4 楼
电话: +86 (756) 3882190
传真: +86 (756) 3882193
电邮: info@buildwin.com.cn

建荣半导体(深圳)有限公司

广东省深圳市福田区深南大道 6013 号有色大厦 922 室
电话: +86 (755) 25334930 / 25334970
传真: +86 (755) 25334931
电邮: sales@buildwin.cn

网站:
http://www.appotech.com
http://www.buildwin.com.cn